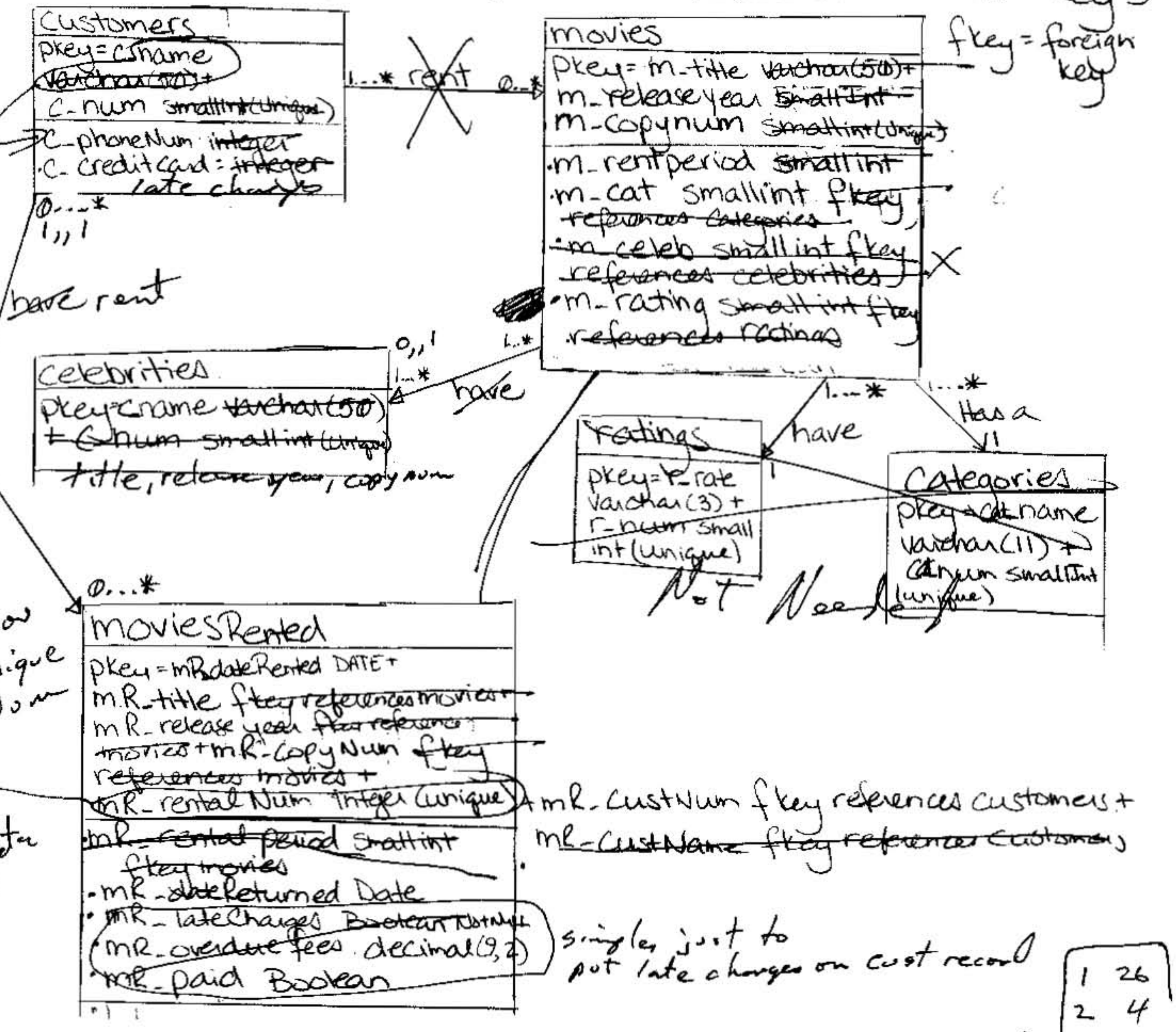
26 Name

1. Entity Relationship diagram (worth 40 marks)

Carefully consider the requirements identified in the case study and translate them into a detailed entity-relationship diagram of a database that meets these requirements. You may also want to consider the requirements of parts 2 and 3 of this exam. Do not include any unnecessary entities or attributes. Your diagram should include:

- The entities that need to be represented by tables in the database
  - Each entity should be represented by a box that has three sections that contain:
    - A suitable name for the entity
    - The attributes that will be used as the primary key of the entity
    - The other attributes that need to be included in the entity

    (I know that I said that so much information in a diagram can be hard to read, but it's easier for you to answer the question this way)
- The relationships that need to be represented in the database
  - Each relationship should be named meaningfully
  - Each end of a relationship should include multiplicity information.



pkey = primary key
fkey = foreign key
Customers
pkey = c_name
~~varchar(50)~~ +
c_num ~~smallint(Unique)~~
c_phoneNum: ~~integer~~
c_creditcard: ~~integer~~
late charges
1...* ~~rent~~ 0...*
0...*
1,,1
have rent
movies
pkey = m_title ~~varchar(50)~~ +
m_releaseyear ~~SmallInt~~ +
m_copynum ~~smallint(Unique)~~
m_rentperiod ~~smallint~~
m_cat smallint ~~fkey references categories~~
~~m_celeb smallint fkey references celebrities~~
m_rating ~~smallint fkey references ratings~~
0,,1
1...*
1...*
have
celebrities
pkey = cname ~~varchar(50)~~
~~+ c_num smallint (unique)~~
~~title, release year, copy num~~
1...*
have
1
ratings
pkey = r_rate
varchar(3) +
~~r_num small int (unique)~~
1...*
Has a
1
~~Categories~~
~~pkey = cat name~~
~~varchar(11) +~~
~~cat num smallint (unique)~~
Not Needed
0...*
moviesRented
pkey = mR_dateRented DATE +
mR_title ~~fkey references movies~~ +
mR_release year ~~fkey references movies~~ + mR_CopyNum ~~fkey references movies~~ +
mR_rental Num integer (unique)
~~mR_rental period smallint fkey movies~~
mR_dateReturned Date
mR_lateCharges ~~Boolean NotNull~~
mR_overdue fees decimal(9,2)
mR_paid Boolean
+ mR_CustNum fkey references customers +
~~mR_CustName fkey references customers~~

Either you use a unique rental Num OR use the other date as a PK

simpler just to put late charges on cust record

1	26
2	4
3	[illegible]
Total	[illegible] 3

You have cluttered this diagram with much data that is not required in the question

4

Name

2. SQL data definition (worth 14 marks)

You are to provide a set of SQL data definitions (as required in the subsections of this question) relating to the main assignment table only. Do not create any unnecessary definitions or parts of definitions.

- It is expected that you can determine which table I mean from the case study and your e-r diagram. The main assignment table is the first table into which some information about a new assignment is entered.
- Be sure to use suitable naming of all components as discussed in class.

a. Provide the SQL required to define the domains you will use in your definition of the main movie table.

- CREATE DOMAIN nums AS SmallInt;
- a CREATE DOMAIN uniqueNum AS SmallInt UNIQUE; ✗
  OR
  a CREATE DOMAIN uniqueNum AS serial; no postgresql.

✓ • CREATE DOMAIN title AS varchar(50);

This would assign a unique # to every movie in the store

This needs to be constrained to 1-14 by a domain

b. Provide the SQL required to define the main movie table. I used the domains I created above)

✓CREATE TABLE movies (✓m_title title, m_release year nums, m_CopyNum uniqueNum, m_rentperiod nums, ✗m_cat nums, ~~m_celeb nums~~, ✗m_rating nums, ✓PRIMARY KEY (m_title, m_releaseyear, m_copyNum), Foreign Key (m_cat) references categories(c_num), FOREIGN KEY (m_celeb) References celebrities(c_num), FOREIGN KEY (m_rating) References ratings (r_num));

if you do This Then you don't need m_title, m_release year in the key since they would be redundant

These should be specified values

2 [redacted] Name

3. SQL Data Manipulation (worth 16 marks)

In question 2 you did not define all of the data definitions you need to answer the following questions. You should use those definitions you have that apply from question 2. However, by using good attribute and view names that relate to the names you used in question 1, it should be obvious what you mean.

[redacted] 2

c. Provide the SQL required to get information from the main movie table about all movies where "Jackie Chan" is known to be an actor. The information should be sorted based on year of release.

Need specific fields

```
+ SELECT  *  FROM movies WHERE
x (m_celeb = (SELECT c-num FROM celebrities
   WHERE (c_name = 'Jackie Chan')))
ORDERBY m-releaseyear;
```

need to allow multiple celebrities per movie

~~* I'm assuming that you don't want the release year displayed even though we are using it to order the movies.~~

Ø

d. Provide the SQL for a subquery that can be used within ~~other queries~~ to find if any copies of a given movie (identified in the main query) are available to be rented right now.

, dm

```
SELECT * FROM movies WHERE (m-title = '<desired movie>'
AND (SELECT m_copynum from movies where (m-title='dm')
NOT IN (SELECT mR-copyNum FROM moviesRented
where (mR-dateReturned >= 'todays date')))) ;
```

what about release year

This would not work because you would not ever have a value in date returned for movies not yet returned. Thus date Returned is never > today's date

putting today's date in quotes makes it a litteral which cannot be compared with a date which is numeric